Experiment Scenarios

Massimo Paolucci, Bertrand Souville, Gordon Blair, Paul Grace, Trân Huynh, Guillaume Tuloup, Hugues Vincent, Antoine Léger, Animesh Pathak, Nikolaos Georgantas, et al.

CONNECT

Emergent Connectors for

Eternal Software Intensive Networked Systems

ICT FET IP Project

Deliverable D6.1

Experiment scenarios

SEVENTH FRAMEWORK PROGRAMME

http://www.connect-forever.eu

INRIA docomo DOCOMO Euro-Labs LANCASTER UNIVERSITY THALES Universita' dell'Aquila technische universität dortmund UNIVERSITY OF OXFORD UPPSALA UNIVERSITET

Project Number	:	231167
Project Title	:	CONNECT – Emergent Connectors for Eternal Software Intensive Networked Systems
Deliverable Type	:	Report

Deliverable Number	:	D6.1
Title of Deliverable	:	Experiment Scenarios
Nature of Deliverable	:	R
Dissemination level	:	PU
Internal Document Number	:	
Contractual Delivery Date	:	February 1st, 2010
Actual Delivery Date	:	February 15th, 2010
Contributing WPs	:	WP6
Editor(s)	:	Trân Huynh
Author(s)	:	Massimo Paolucci & Bertrand Souville (DOCOMO)
		Gordon Blair & Paul Grace (LANCASTER)
		Trân Huynh & Guillaume Tuloup & Hugues Vincent & Antoine Léger (THALES)
		Animesh Pathak & Nikolaos Georgantas & Valérie Issarny & Amel Bennaceur & Rachid Saadi (INRIA)
		Romina Spallazzese & Massimo Tivoli & Patrizio Pelliccione & Massimo Tivoli (UNIVAQ)
Reviewer(s)	:	Bertrand Souville
		Bernhard Steffen
		Paul Grace
		Animesh Pathak
		Nikolaos Georgantas

Abstract

The scenarios proposed in this deliverable are intended to identify the multiple dimensions of interoperability, and also foster further discussions with other work packages (WP1 to WP5). These will be refined and extended as the project progresses. The scenarios have been grouped into three computing domains: *Ubiquitous computing*, *Daily life support* and *Telco Web 2.0 & Cloud Computing*. Although the scenarios are partitioned into domains, they all have common properties such as *the presence of disparate devices*, *a large proportion of peer-to-peer traffic* and *disparate interaction patterns*.

Document History

Version	Type of change	Author(s)
1.0	Initial release.	
1.1	Template Revision to 1. Bring all description items together 2. Highlight interoperability problems	B. Souville, M. Paolucci – DOCOMO
1.2	Cosmetic update.	T. Huynh – Thales
1.3	Renamed section titles.	T. Huynh – Thales
1.4	Added Lancaster, DOCOMO, Thales and INRIA scenarios.	P. Grace – Lancaster B. Souville, M. Paolucci – DOCOMO A. Leger, G. Tuloup, H. Vincent, T. Huynh – Thales A. Bennaceur. N. Georgantas, V. Issarny, A. Pathak, R. Saadi – INRIA
1.5	Updated introduction. Added a summary table of the interoperability issues stated in the experiment scenarios. Removed network interaction level and moved the network topology to the description of the scenario.	B. Souville – DOCOMO P. Grace – Lancaster A. Pathak – INRIA T. Huynh – Thales
1.6	Updated interoperability challenges summary table, introduction and some Mapping to CONNECT issues sections. Added conclusion. Fixed typos.	P. Grace – Lancaster N. Georgantas – INRIA T. Huynh – Thales
1.6.1	Added document review data.	T. Huynh – Thales
1.6.2	Fixed typos.	T. Huynh – Thales
1.6.3	Added missing authors. Repagination.	T. Huynh – Thales
1.6.4	Repagination.	T. Huynh – Thales

Document Review

Date	Version	Reviewer	Comment
27/01/2010	1.4	B. Souville – DOCOMO	Introduction needs rework. Add a summary table of the interoperability issues. Add a list of references.
31/01/2010	1.4	B. Steffen – TUDO	Weak descriptions of the issues highlighted by the scenarios with regards to CONNECT. Missing conclusion.
2/02/2010	1.5	N. Georgantas – INRIA	Confusing terminology used throughout the document. Suggestion for the rework of the introduction.
2/02/2010	1.5	P. Grace – Lancaster	Reworked the summary table of the interoperability issues. English proof-reading and correction.
15/02/2010	1.6	V. Issarny – INRIA	Missing document review data.

Table of Contents

1 Introduction

Scenarios play a major role in understanding the interoperability issues arising in real-world situations, and thus are extremely important to the research carried out in the CONNECT project. We believe that the following conditions should necessarily be present in any scenarios for which CONNECT will be relevant. Namely,

1. **Presence of disparate devices.** The devices present in the system should not have been designed together with interoperability with each other in mind. Preferably, the devices should come from different manufacturers, if not from different countries. This will lead to a high probability of the existence of challenges that CONNECT aims to solve.
2. **Large proportion of peer-to-peer traffic.** A system in which most of the traffic goes through the Internet can be "centrally controlled", and hence the CONNECT capabilities might not be needed as compliance to the Web standards would help coping with the interoperability issues. While it is unfair to assume *zero* internet access from the devices involved, a large proportion of interactions in these systems should ideally be directly between the devices involved. This can be for the purposes of increasing bandwidth, or saving energy, for example.
3. **Disparate interaction patterns.** A system where there is only one type of interaction between components (e.g., a distributed database) is much more homogenous than what we want to address. We believe that the presence of other idioms such as address-and-control, and automatically-monitor-and-alert adequately enrich the problem scenario.

In this deliverable, we present a set of scenarios that have each been contributed by a particular project partner. The purpose of these scenarios is twofold. Firstly, to discover the interoperability challenges of complex applications that the CONNECT project will address. We show that the scenarios identify: i) application-level, ii) interaction protocol, iii) data and iv) non-functional properties interoperability as the important dimensions of interoperability that CONNECT will resolve to enable eternal interoperability within dynamic systems. Secondly to foster further discussions between the other work packages (WP1 to WP5) in order to achieve the necessary integration of expertise for the CONNECT project to proceed; deliverable D1.1 describes how the distributed marketplace scenario has been used for this purpose (1).

The scenarios have been grouped into three scenario families: *Ubiquitous computing*, *Daily life support* and *Telco Web 2.0 & Cloud Computing*. All scenarios use a common template which is structured as follows:

1. **Description**: In this first section, the scenario is described in details and all actors involved in the scenario and the roles they are playing are defined.
2. **Interoperability issues highlighted by the scenario**: We have defined so far three dimensions of interoperability issues:
    - At the network level where a mismatch occurs in the network protocols
    - At the interaction protocol level where a behavioral mismatch among protocols run by interacting parties takes place
    - At the data level where different data or semantic models are used by interacting parties
3. **Mapping to CONNECT issues**: Finally, this section defines a set of requirements which are or shall be considered in the other work packages (i.e. learning, connector synthesis, dependability and security requirements).

The scenarios themselves will be refined and extended as the project progresses to highlight the new and important directions that the CONNECT solutions address.

2 Ubiquitous computing

2.1 Scenario: Flood Prediction and Monitoring

Contributors: Paul Grace, Gordon Blair

2.1.1 Description

Sensor networks are deployed in rivers, estuaries and bays to monitor temperature, water level, flow rate, and pollution. These sensor networks collect raw data that is then transferred to modelling tools (these run on devices with significant computational resources e.g. cluster computers, or computational Grids), which then model environmental conditions that can be used to predict adverse situations in the future e.g. predicting flooding.

The scenario involves a group of scientists who connect at runtime a set of sensor networks that will feed into their current models i.e. at runtime they find sensor networks that collect data (in areas of interest) and connect them to their prediction software.

2.1.1.1 List of Actors

Actor	Type	Localization	Role
Scientist	Human		Input requirements and receive models
Flood Modelling	Service	URL	Process data and predict flood
Sensor Network 1	Service	URL	Collect data
Sensor Network 2	Service	URL	Collect data

2.1.1.2 Storyboard

Step #	Actor	Event
1. Search for available sensor networks in regions of interest.	Scientist	List of sensor network description and endpoints
2. Start Modelling Service	Scientist	
3. Connect sensor networks to Modelling service (using chosen list of 2 sensor networks)	Scientist	
4. Periodically send data to Modelling Service	Sensor Network 1,2	Depth data, Flow rate data
5. Process data	Modelling Service	Depth model, Flow model
6. If flood 'Create' prediction	Modelling Service	New flood prediction

2.1.1.3 Network Topology

Figure 2-1 illustrates that the two sensor networks each employ different networking protocols to connect their sensors i.e. Zigbee (2) and Bluetooth (3). Each has a sink node which collects

all content from the network and is addressable to allow third parties to receive the data. Each sink uses a different networking protocol i.e. GSM and GPRS to communicate with third-parties.

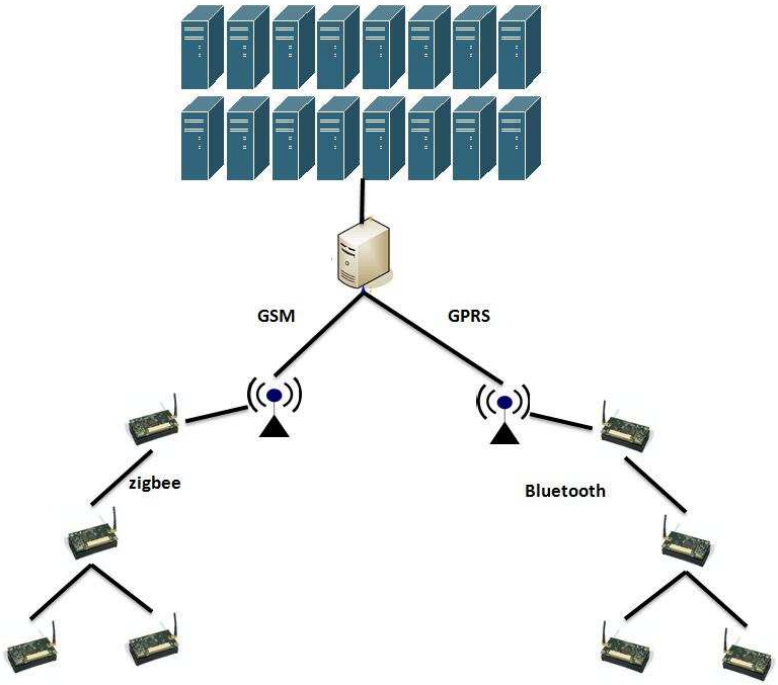

Figure 2-1: Network topology of two sensors connected to a cluster computer

2.1.2 Interoperability Issues Highlighted by the Scenario

2.1.2.1 Interaction Protocol Interoperability

One sensor network employs an event based platform to collect and report data e.g. GEM (4). For this, the sink node collects events that are periodically reported by all the nodes in the network e.g. every node measures depth and flow rate every 2 minutes and communicates this data. The sink node aggregates this data and sends it to users who are listening for events about flooding. Alternatively, the second sensor network uses a data query middleware platform e.g. Tiny DB (5). In this case, a query is sent requesting a current report from the network e.g. the average water depth in the network. The scientist's modelling software is developed upon an event-based platform, in this case SIENA (6); that is, it subscribes for flooding events which are then input to the executing modelling service as they are received.

There are two levels of interoperability challenge here:

- Protocol heterogeneity. Where the two protocols behave the same e.g. the two publish-subscribe systems (SIENA and GEM) they still cannot interoperate because the message formats and event description languages for both systems are different.
- Behaviour heterogeneity. The modelling service is passive i.e. it waits to receive events of interest. However, the sensor network using Tiny DB is active; this means that a request must be sent periodically to receive data back. If data is expected every 2 minutes then a data query must be sent every 2 minutes. The protocol heterogeneity problem above remains for these two systems.

2.1.2.2 Data Interoperability

The scientists modelling software is operating using metric measurements; however the sensor networks are located in the USA. The depth measures are returned in feet, whereas the input is expected in metres. Similarly for flow rate, the expected measure is cubic metres per second while the American sensor network reports cubic feet per second.

2.1.3 Mapping to CONNECT Issues

2.1.3.1 Learning Requirements

Sensor networks need to be discovered prior to the collection of data.

2.1.3.2 Connector Synthesis Requirements

The scenario highlights the key CONNECT issues:

- Diverse communication protocol behaviour. Interoperability between protocols with conflicting behaviour e.g. active and passive protocols remains an unresolved issue.
- Composition of heterogeneous elements into a complex system of systems is underpinned by the fundamental requirement that they be able to interoperate.
- Synthesis of the Event-to-Event connector and synthesis of the Request-to-Event connector must be performed dynamically.

2.1.3.3 Dependability and Security Requirements

This scenario does not highlight dependability or security requirements.

2.2 Scenario: Road Tunnel Accident

Contributors: Paul Grace, Gordon Blair

2.2.1 Description

The scenario describes how a fire in a road tunnel occurs, and how emergency response to the scenario is deployed. The following is taken from the EU RUNES project (7) to set the scene:

> *"On a busy weekday a collision occurs deep within the tunnel between several vehicles including a tanker loaded with vegetable oil that suffers penetration of the tank and begins to leak over the road surface. A small fire started as a result of the collision spreads to the oil, which begins to burn producing clouds of thick smoke as well as heat and flame."*

Upon detection of the fire an emergency response team travels to the tunnel. They communicate with one another using mobile devices connected using ad-hoc networking technology. A leader co-ordinates the fighting of the fire (based upon information about the environmental conditions in the tunnel; and co-ordinates the rescue of trapped road users.

Vehicles trapped in the network have networking capabilities and employ vehicular ad-hoc network technologies to communicate with other vehicles; this network can be used to inform

(and receive directions from) the response team about location in the tunnel and the current status.

The tunnel is equipped with a sensor network that senses temperature.

2.2.1.1 List of Actors

Actor	Type	Localization	Role
Vehicle Road User	Human		Be rescued
Controller	Human		Co-ordinate firefighters, retrieve information about fire and trapped persons
Firefighter	Human		Put out fire, rescue
Tunnel Sensor Network	Service	URL	Sense temperature and report periodically

2.2.1.2 Storyboard

Step #	Actor	Event
1.	Controller/Firefighter	Arrive at tunnel
2.	Controller	Connect to Tunnel Sensors
3.	Controller	Request sensor data
4.	Controller	Connect to tunnel VANET
5.	Controller	Connect to Firefighters MANET
6.	Controller	Send message to VANET with info and request for response
7.	Vehicle Road User	Respond to controller
8.	Controller	Direct firefighters
9.	Firefighter	Put out fire and rescue as directed

2.2.1.3 Network Topology

The VANET uses dedicated short range communication (DSRC) to connect the cars. The sensor network is connected via Zigbee and the sink node is addressable by third parties using Zigbee. The mobile devices of the controllers and firefighters are connected using 802.11b in ad-hoc mode. Bridges are required to ensure that there is connectivity between the three types of network. Figure 2-2 demonstrates the topology of the network.

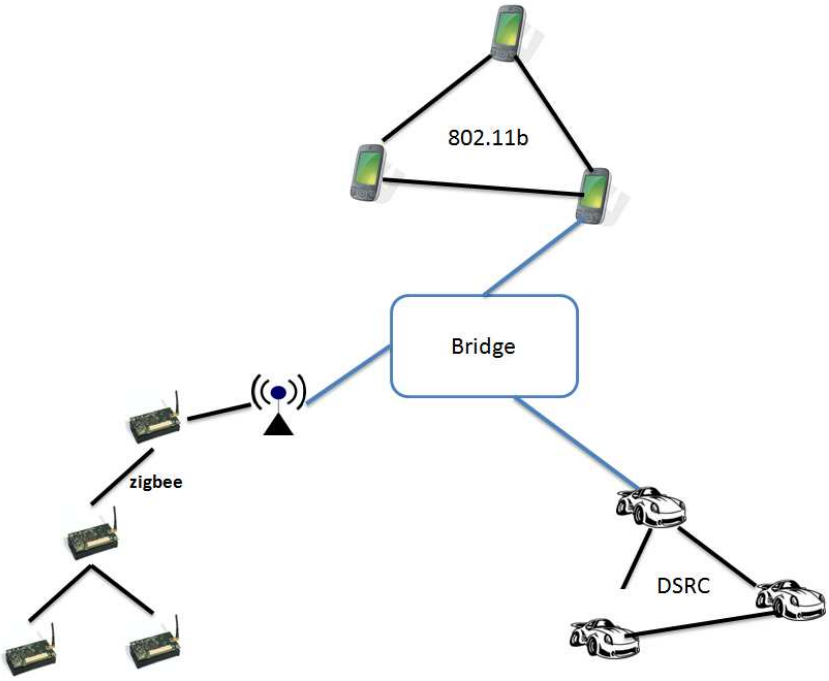

Figure 2-2: Tunnel network topology

2.2.2 Interoperability Issues Highlighted by the Scenario

2.2.2.1 Interaction Protocol Interoperability

The following middleware protocols are employed in each of the three domains (sensors, mobile devices, cars):

- Car. A messaging middleware is deployed atop the BBR (8) routing protocol.
- Mobile device. The Lime tuple space middleware (9) is used to communicate between the controllers and firefighters.
- Sensor Network. The Gem event-based middleware is used to disseminate periodic events about the temperature of parts of the tunnel.

When the controller needs to connect to the sensor network the tuple space and event paradigms conflict; an event must be translated into a tuple that can be read from the space. To connect to the VANET, the mobile devices must be able to receive, route and understand message routed using the BBR protocol.

In these cases, there are protocol message format challenges and protocol behaviour mismatches that must be resolved.

2.2.2.2 Data Interoperability

The application uses standardized SI units e.g. temperature and location (GPS); further, messages are text-based – hence there are no data level interoperability problems.

2.2.3 Mapping to CONNECT Issues

2.2.3.1 Learning Requirements

The scenario is peer-to-peer in nature which cannot be resolved by traditional interoperability solutions. Connections between peers use ad-hoc networking technologies and thus communication protocols need to be learnt for each participating peer.

2.2.3.2 Connector Synthesis Requirements

The three parties are unaware of each other until the emergency scenario occurs; and cannot be planned and designed for in advance. Connector synthesis can only happen when the peer is discovered.

2.2.3.3 Dependability and Security Requirements

This scenario does not highlight dependability or security requirements.

2.3 Scenario: Large gathering of people in a stadium

Contributors: Amel Bennaceur. Nikolaos Georgantas, Valerie Issarny, Animesh Pathak, Rachid Saadi

2.3.1 Description

The scenario which we consider is that of a *large gathering of people in a stadium*, for example, for a concert or watching a football game. The scenario is partly inspired by one of the situations discussed in the work of one of our collaborators[1]. We will proceed with the specific case of a football game in the following text.

To properly simulate a realistic situation in the CONNECT world, we assume that there already exist various implementations of the applications, implemented by various developers in their own countries. However, these are not interoperable at the outset, due to the choices made during the design and development phases of these apps. These interoperability challenges might also arise due to the choice of technology/device used (e.g., iPhones versus Windows Mobile-using HTC Phones versus Motorola Droid phones with Android).

The specific use cases are expressed in later scenarios.

2.3.1.1 List of Actors

Actor	Type	Localization	Role
Stadium	Service	URL, accessible by joining a given wi-fi network	Providing services to all other actors
Security Staff	Human	Inside the stadium. Mobile.	Manage security of the stadium

[1] "Middleware for Mobile Sensing Applications in Urban Environments", PhD Thesis, Oriana Riva, Univ. of Helsinki

Audience/Consumer	Human	Inside the stadium. Mobile but mostly static.	Watch the game, browse and buy items, etc.
Merchant	Human	Inside the stadium. High mobility.	Sell items.

2.3.1.2 Storyboard

Step #	Actor	Event
1	Audience	Enters Stadium. Connects to stadium service.
2	Stadium	Responds to Audience's request for join. Registers in local database.

The above "registration" steps repeat for Security Staff and Merchant actors also, as they enter the stadium.

The later steps are discussed in the following sections on a per-scenario basis. The overall list of applications is as follows

During the course of the game, we would like to enable the following behaviors of the actors discussed above, using the components they possess:

1. Audience members share/exchange digital "trading cards" with each other using their phones.
2. Audience members can share the pictures they have taken of the game with others present in the stadium.
3. A higher-quality version of the pictures above can also be put up for sale if the photographer so desires
4. Audience members can purchase special camera angles of the game to view on their SmartPhones.
5. The security staff can monitor the stadium using the camera, and control them if they want to focus on a specific region.
6. The security staff can be notified of suspicious activity as inferred by the camera-network.
7. Audience members can choose to sign up with a "friend finder" application which broadcasts their location to their friends in the stadium.
8. Audience members can register their intent to purchase certain items (e.g., popcorn), tagged with their location.
9. Merchants can use the "intent" registered above to reach potential consumers in the stadium.
10. Readings from acoustic sensors can be used to adjust audio-levels at individual speakers so as to ensure that all audience members can hear clearly. This becomes especially important if the event involves speeches etc.
11. Audience members should be able to know the occupancy situation of the toilets in the region around them.
12. In case of emergency, e.g., a fire or a structural lack of integrity, individuals should be able to get warning messages guiding them to proper exits. Sensors in exit tunnels can be used to 'even out' the human traffic.

2.3.2 Interoperability Issues Highlighted by the Scenario

In our scenarios, we wish to address interoperability in 4 dimensions

1. **Application Level:** Here, the two developers who implemented the same system used different logic. This might include different ordering of interchangeable operations, or using a single higher-level operation instead of several lower level ones.
2. **Middleware Level:** Here, the systems that need to communicate were designed to use different middleware. This might include one actor using an application based on a publish-subscribe middleware, while the other uses a UPnP+SOAP based middleware.
3. **Platform Level:** Here, we tackle the case where the systems were implemented over different software technologies, and explore the issues arising out of that. A classical example will be an application which has been developed by one developer using Microsoft technologies, while the other developer used Java technology.
4. **Data Level:** Here, the systems that need to communicate use different data formats. A good example would be applications that assume the units of currency to be that of the country they were developed in.

2.3.2.1 Interaction Protocol Interoperability

As discussed above, different implementations of the same application can use different interaction protocols. We discuss the specific options available to applications in the later sections with detailed description of the scenarios.

2.3.2.2 Data Interoperability

As discussed above, different currency formats can be one example of data heterogeneity that might be exhibited by the system. We discuss the specific options available to applications in the later sections with detailed description of the scenarios.

2.3.3 Mapping to CONNECT Issues

The scenarios arising in the football stadium will potentially use one or more CONNECT enablers in order to address the interoperability challenges arising in them. The details are provided in the following sections.

2.4 Scenario: Stadium: Warning System

Contributors: Amel Bennaceur. Nikolaos Georgantas, Valerie Issarny, Animesh Pathak, Rachid Saadi

2.4.1 Description

In the stadium discussed above, the users of the system can subscribe to warnings coming from the system, and then get notifications on their mobile devices once the warning is issued.

2.4.1.1 List of Actors

Actor	Type	Localization	Role
Stadium	Service	URL, accessible by joining a given wi-fi network	Providing services to all other actors
Security Staff	Human	Inside the stadium. Mobile.	Manage security of the stadium

Audience/Consumer	Human	Inside the stadium. Mobile but mostly static.	Watch the game, browse and buy items, etc.
Merchant	Human	Inside the stadium. High mobility.	Sell items.

2.4.1.2 Storyboard

Step #	Actor	Event
1	Audience/Security Staff/Merchant	Subscribes to Warning Event
2	Stadium	Detects a disaster situation (e.g., temperature sensors detect fire)
3	Stadium	Sends out warning message to attendees
4	Audience/Security Staff/Merchant	Receive alert. Take evasive actions.

2.4.2 Interoperability Issues Highlighted by the Scenario

2.4.2.1 Interaction Protocol Interoperability

The stadium warning system uses a Publish-Subscribe middleware protocol, whereas the system on the user's device is based on a SOAP-based middleware. This causes a problem, for publish-subscribe the Stadium simply sends the warning message to the publish-subscribe bridge, which then automatically notifies the other users. However, in the SOAP-based implementation, the users will need to be contacted one-by-one to inform them of the problem.

2.4.2.2 Data Interoperability

We assume no data-level mismatch in this scenario.

2.4.3 Mapping to CONNECT Issues

2.4.3.1 Learning Requirements

If a SOAP-based user enters a stadium with a Publish-Subscribe-based system, learning of the user's protocol will be needed.

2.4.3.2 Connector Synthesis Requirements

The connector synthesis enabler will need to generate the connector for the systems. This might happen by a process in which it uses existing connectors between other sets of interaction patterns, and composes them to provide the connector required in the current scenario.

2.4.3.3 Dependability and Security Requirements

The system will have to measure up to latency requirements, since warning messages cannot be delayed i.e. the interoperability solution must meet the dependability requirements of the stadium and the client for the message to be delivered in the required time.

2.5 Scenario: Stadium: Distributed Marketplace ("Popcorn" Scenario)

Contributors: Amel Bennaceur. Nikolaos Georgantas, Valerie Issarny, Animesh Pathak, Rachid Saadi

2.5.1 Description

In the stadium discussed above, the Merchant can register in the system with the products they are selling, and the Consumer members can then browser this marketplace and place orders. The Merchant can then respond with a yes/no, and if the answer is yes, then the Consumer gets an alert on his mobile device when the Merchant is near. This scenario is discussed at length in Section 2 of D1.1 (1).

2.5.1.1 List of Actors

Actor	Type	Localization	Role
Stadium	Service	url, accessible by joining a given wi-fi network	Providing services to all other actors
Consumer	Human	Inside the stadium. Mobile but mostly static.	Watch the game, browse and buy items, etc.
Merchant	Human	Inside the stadium. High mobility.	Sell items.

2.5.1.2 Storyboard

Step #	Actor	Event
1	Merchant	Registers product with system
2	Consumer	Browses the marketplace for getting the list of products
3	Consumer	Further refines his search to get the list of all Merchants selling a particular product
4	Consumer	Places an order with a specific Merchant for a specific product
5	Merchant	Receives request from Consumer. Responds with a Yes or No to the request.
6	Merchant /System	Sends an alert to the Consumer if the answer was Yes and the Merchant is close enough.

2.5.2 Interoperability Issues Highlighted by the Scenario

2.5.2.1 Interaction Protocol Interoperability

In one instance of the scenario, the Merchant's system is based on SSDP(UPnP) for discovery, and SOAP for request handling. The Consumer's system is based on a LIME implementation of Tuple Spaces. These are very different middleware with completely different underlying system models. While in SOAP, the consumers would be directly interacting with merchants to place orders by passing SOAP messages, in a Tuple Space, all interactions must take place via the shared data store provided by the Tuple Space.

2.5.2.2 Data Interoperability

In this scenario, the Merchant and Consumer may be using different currencies. The connector will be required to perform on-the-fly conversion to ensure that the users are transparent to this heterogeneity.

2.5.3 Mapping to CONNECT Issues

2.5.3.1 Learning Requirements

The learning enabler will need to learn the behavior of the systems used by both the Merchant and the Consumer. The interface description (in form of WSDL) will be available, as well as a live system to interact with.

2.5.3.2 Connector Synthesis Requirements

The connector synthesis enabler will need to synthesize the connector to connect the different middleware being used. This might happen by a process in which it uses existing connectors between other sets of interaction patterns, and composes them to provide the connector required in the current scenario.

2.5.3.3 Dependability and Security Requirements

If the system includes payments, then security issues assume importance in this scenario. For example, different authentication and authorization protocols may be employed by the two peers.

3 Daily life support

3.1 Scenario: Airport Boarding Cards

Contributors: Paul Grace, Gordon Blair

3.1.1 Description

A traveller checks into her flight using her mobile device while on the way to the airport. When she arrives she prints her ticket directly using a nearby printer for boarding cards.

3.1.1.1 List of Actors

Actor	Type	Localization	Role
Traveller	Human	Airport	User
Printer	Service	URL	Printing

3.1.1.2 Storyboard

Step #	Actor	Event
1. Input 'print' on mobile device application	Traveller	Print request
2. Receive a print request	Printer	Print document
3. Collect document from printer	Traveller	

3.1.1.3 Network Topology

Typically, the mobile device and the printer will be network addressable using one or other wireless networks. They could employ infrastructure-based networking e.g. IEEE 802.11b for the printer, and 3G for the phone; alternatively they could employ ad-hoc networking protocols e.g. Bluetooth or 802.11b in ad-hoc mode. This highlights the potential network interoperability that could be encountered when this application is used and deployed in many different locations. In this particular situation the mobile device uses Bluetooth connectivity, while the printer employs 802.11b in ad-hoc mode.

Figure 3-1 illustrates how the mobile device connects to the printer via the network connections available. Because they utilise different network types a gateway device with Wifi and Bluetooth network interfaces is used; this routes messages from Bluetooth network interface of the mobile device to the Wifi network interface of the printer.

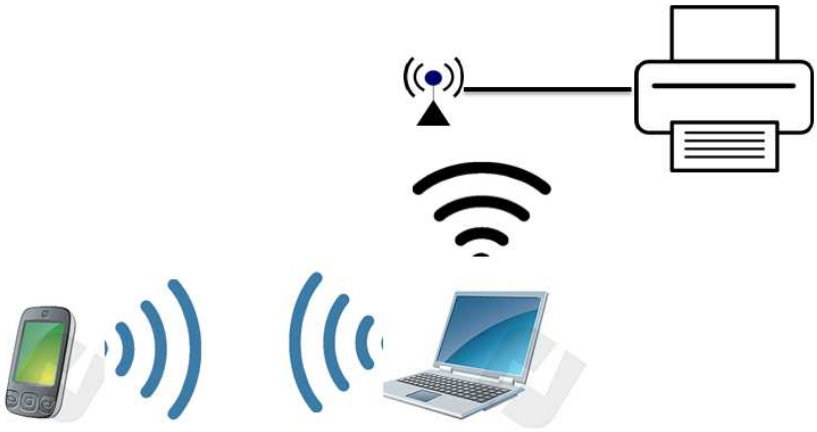

Figure 3-1: Network topology of zero-configuration printing

3.1.2 Interoperability Issues Highlighted by the Scenario

While this appears a simple scenario (that indeed could be implemented to avoid interoperability problems e.g. the avoidance of networked deployments and the use of standard barcodes in real airports), it highlights the significant interoperability challenges that exist in even the basic cases of zero-configuration networking and interaction.

3.1.2.1 Interaction Protocol Interoperability

In the scenario, two types of communication protocols are employed:

- Service Discovery protocols are used by the mobile device to search for a nearby printer to print the boarding card to; and the printer correspondingly uses this type of protocol to advertise its service to prospective users. The Bluetooth mobile device uses SDP (3) to discover networked services; the printer uses Bonjour (10) to advertise.
- Service Interaction protocols are used to send print requests from a computational device to a networked printer. While application printing protocols employ standardised protocols (e.g. lpr (11)) this scenario highlights interoperability at this level too. The mobile device employs the Bluetooth Basic Printing Profile (BPP) (3) which employs the OBEX (Object Exchange) protocol (12) to transfer binary objects to the printer. The wireless printer uses lpr to receive print jobs.

There are significant interoperability challenges within these protocols alone; the mobile device will not be able to discover the advertised printer (because the request will not be received by the Bonjour protocol, nor can it be understood). The printing protocols employed mean that the two cannot systems cannot connect directly (BPP uses OBEX over L2CAP, while LPR uses TCP).

3.1.2.2 Data Interoperability

The mobile device BPP will typically send print requests in the following format:

```
<u:CreateJob xmlns:u="urn:schemas-bluetooth-org:service:Printer:1">
  <JobName>MyJob</JobName>
  <JobOriginatingUserName>mailto:MyEmail</JobOriginatingUserName>
  <DocumentFormat>application/PostScript:3</DocumentFormat>
  <Copies>1</Copies>
  <Sides>one-sided</Sides>
  <NumberUp>1</NumberUp>
  <OrientationRequested>portrait</OrientationRequested>
  <MediaSize> iso_a4_210x297mm</MediaSize>
  <MediaType>cardstock</MediaType>
  <PrintQuality>normal</PrintQuality>
  <CancelOnLostLink>true</CancelOnLostLink>
</u:CreateJob>
```

For each of these data types; there may be a mismatch with what the printer understands by tag e.g. media size and paper size with values iso_a4_210x297mm compared to A4.

3.1.3 Mapping to CONNECT Issues

3.1.3.1 Learning Requirements

Learning occurs when looking for printers to print the boarding pass. The heterogeneity of communication systems from networking protocols, discovery protocols, through communication protocols, to application data encountered when systems spontaneously interact at run-time demonstrates the need for interoperability solutions for such peer-to-peer encounters.

3.1.3.2 Connector Synthesis Requirements

Standards based approaches for ensuring interoperability are involved in this solution (e.g. LPR, Bluetooth, Wifi). However, they exacerbate the problems as they themselves cannot interoperate. This shows that new approaches that do not employ a prior defined legacy standard or middleware are required to make such heterogeneous systems interoperate.

3.1.3.3 Dependability and Security Requirements

This scenario does not highlight dependability or security requirements.

3.2 Scenario: Car Parking

Contributors: Massimo Paolucci, Bertrand Souville

3.2.1 Description

The objective of this use-case is to provide an example that is related to ubiquitous computing, and that still shows interesting data interoperability issues. Specifically, the use case highlights the following two cases:

1. Need to transfer location information in different formats
2. Possible lack of information and the need to ask to the user

A number of European cities are offering SMS parking services. In this scenario, we take Amsterdam and Milan as examples. The parking in both cities requires sending an SMS to pay for parking. In both cases the SMS contains:

1. An identification of the parking spot
2. An id of the car. In the case of Amsterdam this id is the license plate, in the case of Milan the id is provided by the parking authority.

Figure *3-2* show an example of the message to be sent in the case of Amsterdam, and Figure 3-3 shows the car id and parking id for Milan.

Figure 3-2: SMS Message for Amsterdam

Figure 3-3: Parking ID and Car ID in Milan

The problem of sending such an SMS is quite obvious. First it requires the user to enter many different letters and number which is very inconvenient; second, if the user makes a mistake either the payment fails, or he pays for parking of another car in a different place of town.

Yet, there are interesting alternative solutions. The parking id essentially specifies the location where the car is parked. Such a location can be sense via GPS, Cell-id intersection, presence services from the mobile operator or other similar means. Similarly, the car id can be sensed through Bluetooth beacons in the car or through the starting lock for NFC (Near Field Communication) started cars.

The parking process becomes the following:

1. C = get car Bluetooth beacon
2. L = Sense location at parking
3. C_{id} = Transform C in car id,
4. P_{id} = parking id identified from location
5. Send SMS with content C_{id} and P_{id}

A complication may emerge when the user does not have a sensor to sense either one of the two parameters. Then the connecting system needs to recognize the problem and ask the user for the correct parameter.

3.2.1.1 List of Actors

Actor	Type	Localization	Role
Location sensor	Sensor	User Mobile	Sense location

		Phone	
Bluetooth sensor	Sensor	User Mobile Phone	Sense car id
Parking system	System	Unknown	Verify valid parking
Location/Car Connector	Connectors	Unknown	Perform data/protocol mismatches.

3.2.1.2 Storyboard

Step #	Actor	Event
1.	**Bluetooth sensor**	Sense the car and extract car id for different systems
2.	**Location sensor**	Monitor the location of the user. Specifically it is able to say where the user is at the time of parking
3.	**Location Connector**	Map location into parking lot id
4.	**Car Connector**	Select appropriate car id
5.	**Parking system**	Verify the parking of the user

3.2.1.3 Network Topology

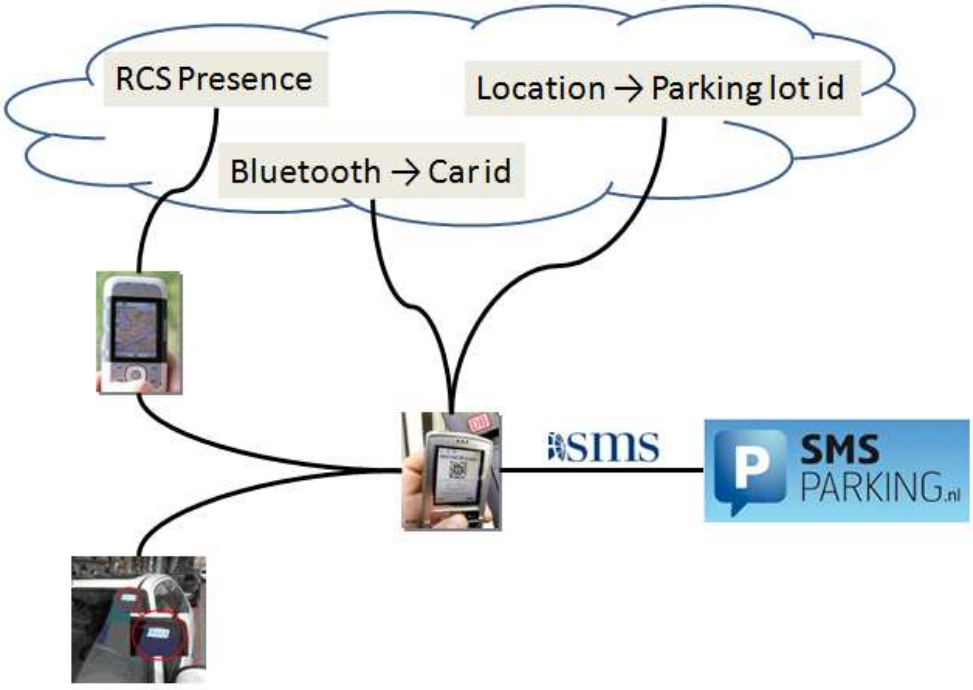

Figure 3-4: Network Topology for the SMS Use Case

The network topology is shown in Figure 3-4 which highlights the different components as well as the connectors and information coming from the network (or from the cloud).

3.2.2 Interoperability Issues Highlighted by the Scenario

This scenario identifies mostly data interoperability issues. First there is a need to transform a location id in the parking id; second there is a need to transform the car id into the correct id required by the parking authority. The latter id may be implemented as a simple table.

3.2.2.1 Interaction Protocol Interoperability

None

3.2.2.2 Data Level Interoperability

As pointed out above there are two types of data interoperability problems:

1. Transform a location id in the parking id;
2. Transform the car id into the correct id required by the parking authority.

The latter task may be implemented as a simple table which specifies the car id given information such as the beacon.

The first issue is more complex because it requires a representation of the location of the parking lot in some coordinate structure and then potentially a second mapping from the coordinate system used by the mobile to the coordinate system used by the mapping.

3.2.3 Mapping to CONNECT Issues

3.2.3.1 Learning Requirements

None

3.2.3.2 Connector Synthesis Requirements

It requires the composition of two connectors: first, the GPS to parking id; the second, car Bluetooth beacon to car id.

3.2.3.3 Dependability and Security Requirements

None at the connector level. There is a problem of guaranteeing that the correct charging for the correct car at the correct location is done.

3.3 Scenario: Card checking

Contributors: Antoine Léger, Guillaume Tuloup, Hugues Vincent, Huynh Ngoc Châu Trân

3.3.1 Description

This scenario illustrates a common action when entering the public transport: card checking. Instead of buying a ticket for each travel, daily commuters have the possibility to subscribe to a season ticket which allows him to take the bus, the tram or the train every day with a single card. The subscription information is stored on the card and it must be checked when using transportation.

The use case is more precisely focused on card checking with the devices found on the RATP (Régie Autonome des Transports Parisiens) buses. With such a device, card checking consists in presenting the card in front of the card reader to validate subscription information. Concentrator and ticketing system are supposed to be connected constantly.

3.3.1.1 List of Actors

Actor	Type	Localization	Role
User	Human		Card owner
Card checker	Device	Bus	Subscription validation
Concentrator	Computer	Bus	Storage
Ticketing system	Computer		Transactions recorder

3.3.1.2 Storyboard

Step #	Actor	Event
1.	User	The user submits the card to the card checker.
2.	Card checker	The card checker reads the card data.
3.	Card checker	The card checker validates the data.
4.	Concentrator	The card information is checked against the blacklist.
5.	Concentrator	The subscription information is validated.
6.	Ticketing system	The subscription information is validated.
7.	Concentrator	The travel is recorded.
8.	User	If the card is rejected, user is not allowed to use the public transport.
9.	Card checker	If the card is validated, card checking operations are over.

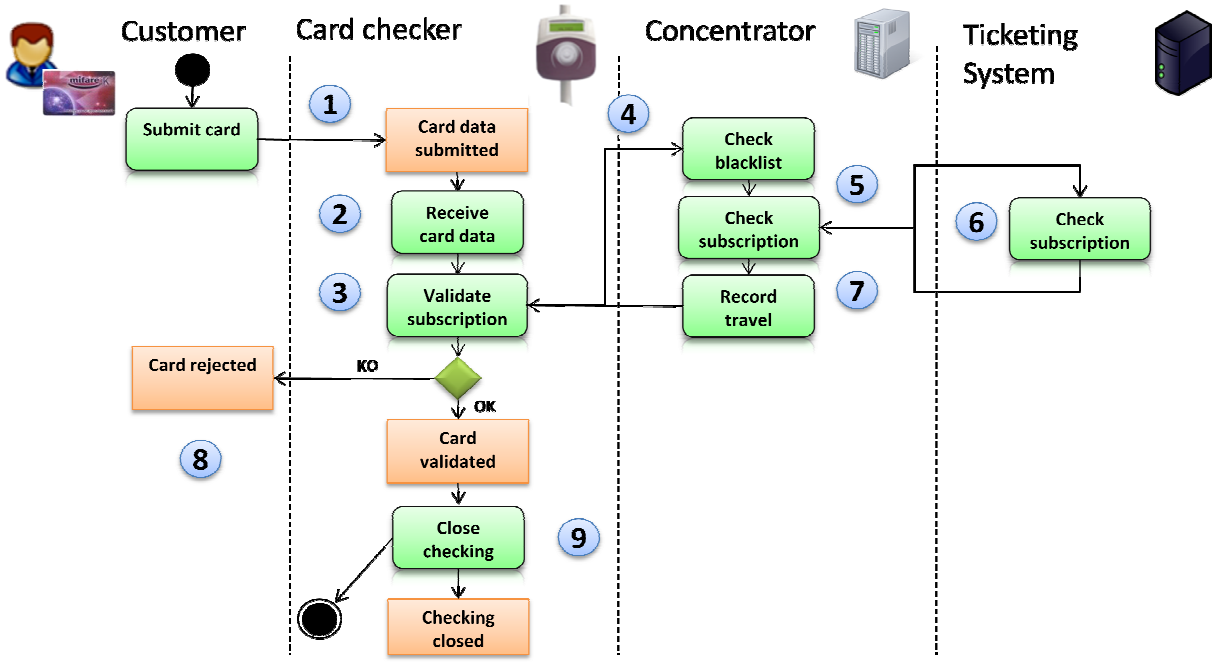

Figure 3-5: Card checking sequence diagram

3.3.1.3 Network Topology

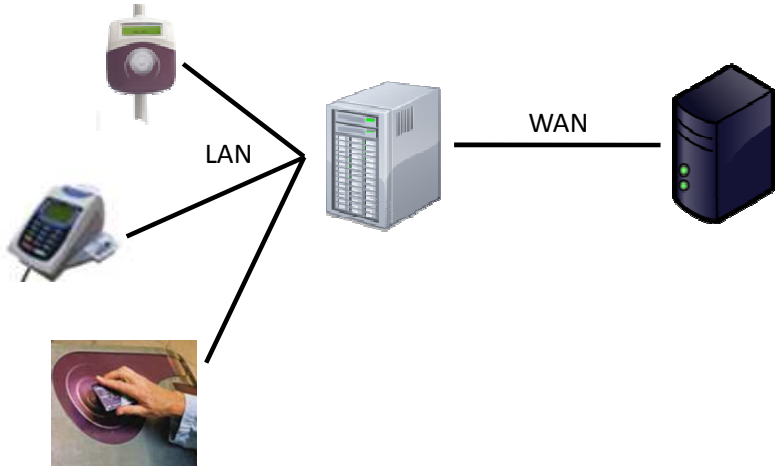

Figure 3-6: Network topology of the card checking system

3.3.2 Interoperability Issues Highlighted by the Scenario

3.3.2.1 Interaction Protocol Interoperability

The concentrator manages a network of RFID (13) card readers, defining the type of cards to index and the actions to take. Protocol interoperability is required to allow communication between the reader network controller and the readers on an IP network.

3.3.2.2 Data Interoperability

This scenario stresses the data interoperability issue. Data from the card readers need to be adapted and transformed into the concentrator exchange format.

3.3.3 Mapping to CONNECT Issues

3.3.3.1 Learning Requirements

There is a need for the connectors to adapt RFID protocols to the IP protocol.

3.3.3.2 Connector Synthesis Requirements

This scenario requires mediating connectors between the card readers and the concentrator.

3.3.3.3 Dependability and Security Requirements

This scenario does not highlight dependability or security requirements.

3.4 Scenario: Card order

Contributors: Antoine Léger, Guillaume Tuloup, Hugues Vincent, Huynh Ngoc Châu Trân

3.4.1 Description

This scenario describes the process of the subscription to a season ticket. It assumes that the card is sent to the customer before he completes the payment. The card manufacturer is responsible for initializing the card data based on the information given in the order.

3.4.1.1 List of Actors

Actor	Type	Localization	Role
Customer	Human		Card buyer
Order systems	Computer		Card order
Card manufacturer	Device		Card manufacturer
Accounting system	Computer		Account manager

3.4.1.2 Storyboard

Step #	Actor	Event
1.	Customer	The customer orders a new card.
2.	Order systems	The order systems receive the card order.
3.	Order systems	The order systems validate the order.

4.	Customer	If the card order is not validated, the order is rejected.
5.	Order systems	If the card order is validated, the card manufacturing starts.
6.	Card manufacturer	The card is manufactured.
7.	Card manufacturer	Based on the order information, the card is initialized.
8.	Card manufacturer	The new card is delivered.
9.	Accounting system	An invoice is sent to the customer.
10.	Customer	After the invoice reception, the customer makes the payment.
11.	Accounting system	The payment is received.
12.	Order systems	The order is closed.

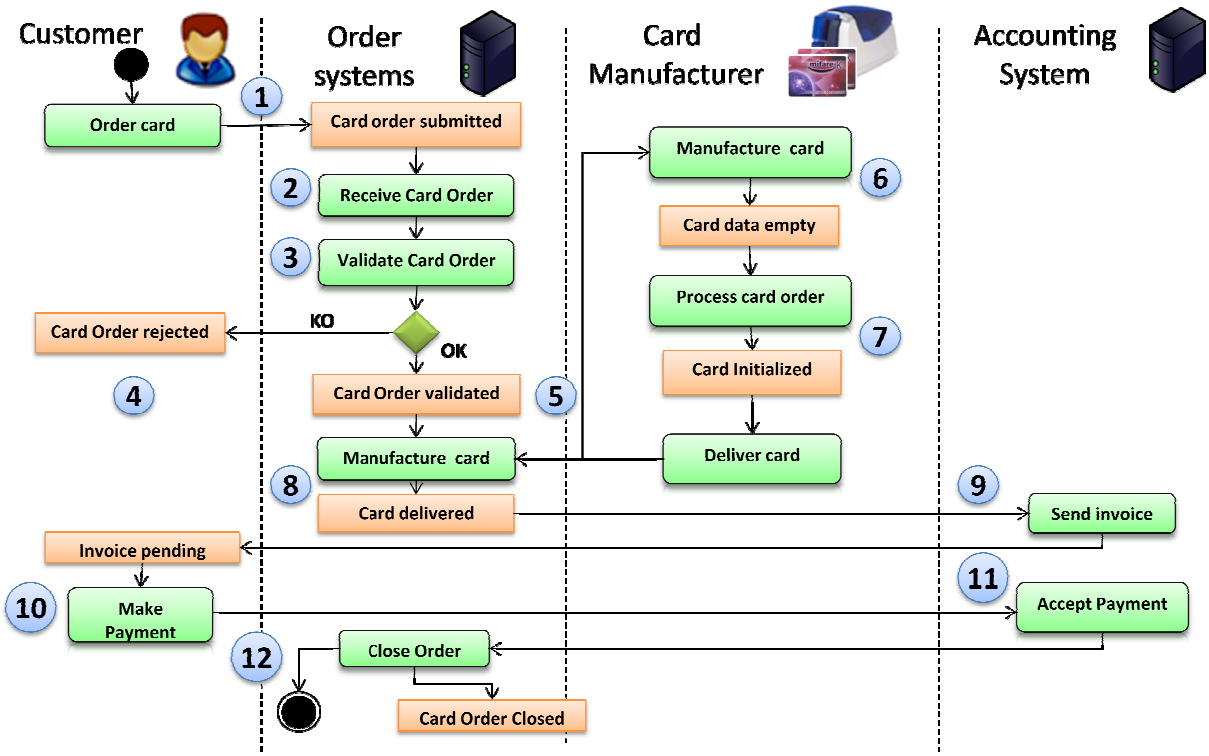

Figure 3-7: Card order sequence diagram

3.4.1.3 Network Topology

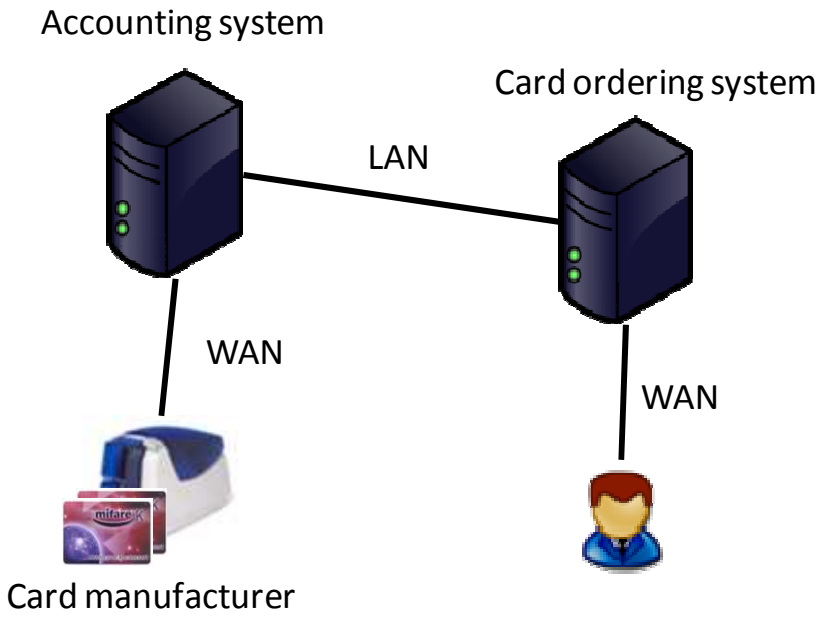

Figure 3-8: Network topology of the card ordering system

3.4.2 Interoperability Issues Highlighted by the Scenario

3.4.2.1 Interaction Protocol Interoperability

The communications between the actors use various protocols depending on the age of the applications that run on the Accounting system and the Card ordering system. Communications are not trivial as there is a mix of protocols like: IIOP (14), REST (15) and SOAP (16).

3.4.2.2 Data Interoperability

This scenario also stresses data and semantic interoperability issues. It cannot be assumed that all the systems use the same data models. The information provided by the customer is very likely stored in different database schemas by the Accounting and the Card ordering systems as well as the Card manufacturer.

3.4.3 Mapping to CONNECT Issues

3.4.3.1 Learning Requirements

There is no need for learning as service descriptions are already exposed (WSDL).

3.4.3.2 Connector Synthesis Requirements

This scenario requires mediating connectors between each actor as they all use different communications protocols as well as different data formats.

3.4.3.3 Dependability and Security Requirements

The information about the customer are sensible data. Security is required to ensure that only authorized and authenticated actors can access them.

3.5 Scenario: Ticket order

Contributors: Antoine Léger, Guillaume Tuloup, Hugues Vincent, Huynh Ngoc Châu Trân

3.5.1 Description

For a single travel, a user needs a ticket to take the train, the tram or the bus. This scenario assumes that the user orders a ticket through the CCHS (Central Clearing House Systems) and eventually withdraws it at a sales device in a station.

3.5.1.1 List of Actors

Actor	Type	Localization	Role
Customer	Human		Ticket buyer
CCHS	Computer		Ticket order
Ticketing systems	Computer		Card manufacturer
Banks	Computer		Account manager
Sales device	Device	Station	

3.5.1.2 Storyboard

Step #	Actor	Event
1.	Customer	The customer orders a ticket.
2.	CCHS	The ticket order is submitted to the CCHS.
3.	CCHS	The CCHS validates the order.
4.	Ticketing systems	The ticketing systems check the timetables.
5.	Ticketing systems	The ticketing systems check the bookings
6.	Ticketing systems	The order is validated against the previous information.
7.	Customer	If the order is not validated, the order is rejected.
8.	CCHS	If the order is validated, an invoice is sent to the customer.
9.	Customer	The customer makes the payment.
10.	Bank	The customer's account is updated accordingly.
11.	Customer	The customer withdraws the ticket.
12.	Sales device	The device prints the ticket.
13.	CCHS	The ticket order is closed.

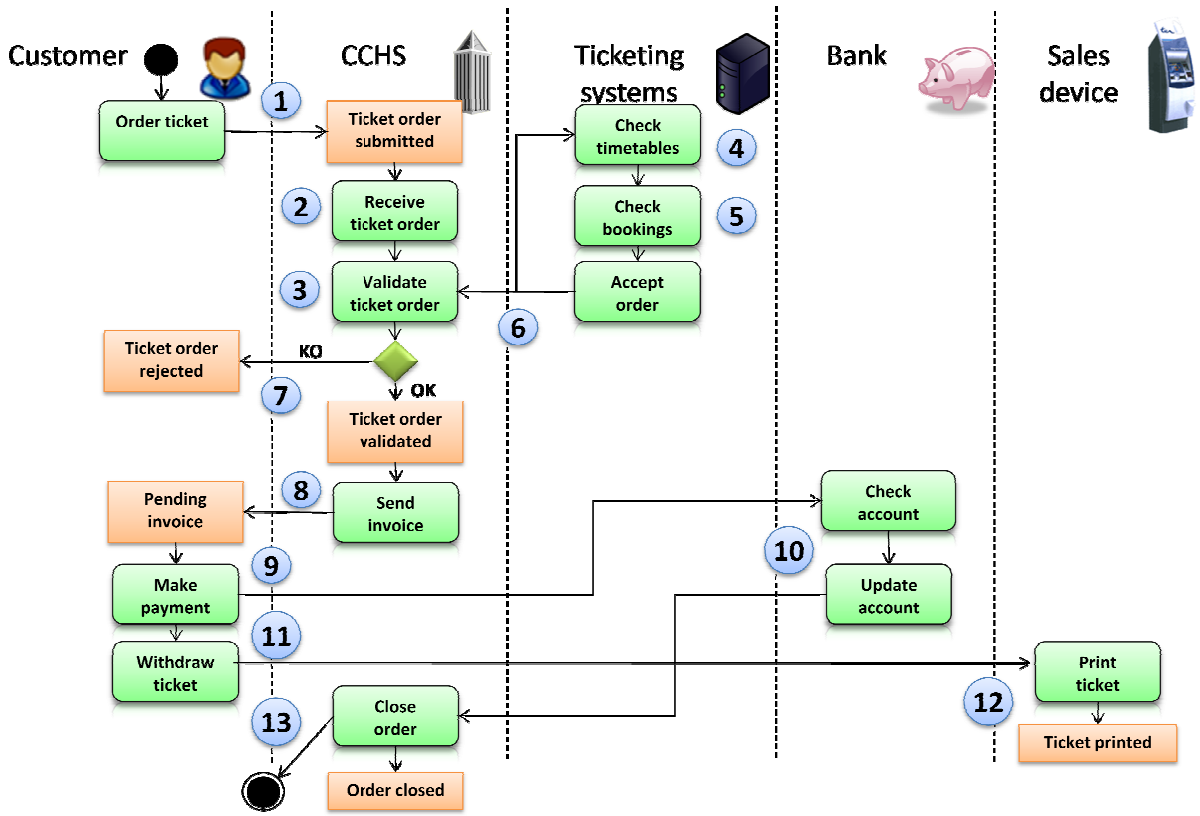

Figure 3-9: Ticket order sequence diagram

3.5.1.2.1 Network Topology

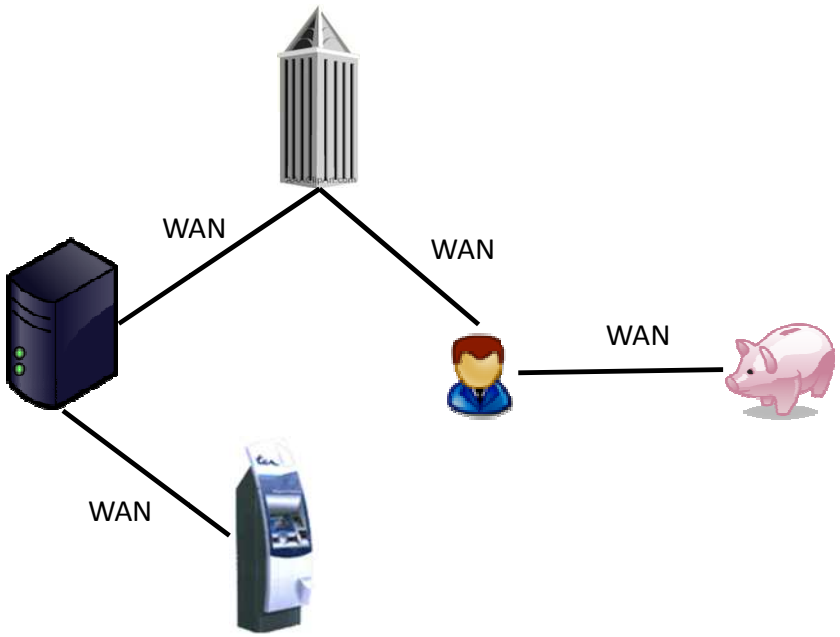

Figure 3-10: Network topology for the ticket ordering system

3.5.2 Interoperability Issues Highlighted by the Scenario

3.5.2.1 Interaction Protocol Interoperability

Various communication protocols are used in this scenario. Applications that interact with the customer will preferably use REST or SOAP message or basic HTTP forms. Communications between the servers will preferably use SOAP or JMS (17).

3.5.2.2 Data Interoperability

This scenario also stresses data and semantic interoperability issues. It cannot be assumed that all the systems use the same data models. There could be as much data models as the number of actors.

3.5.3 Mapping to CONNECT Issues

3.5.3.1 Learning Requirements

There is no need for learning as the services descriptions are already exposed (WSDL).

3.5.3.2 Connector Synthesis Requirements

This scenario requires mediating connectors between each actor as they all use different communications protocols as well as different data formats.

3.5.3.3 Dependability and Security Requirements

The information about the payment are sensible data. Security is required to ensure that only authorized and authenticated actors can access them.

3.6 Scenario: Fire emergency

Contributors: Antoine Léger, Guillaume Tuloup, Hugues Vincent, Huynh Ngoc Châu Trân

3.6.1 Description

This scenario deals with an emergency situation, in particular a fire emergency. When facing such a situation, typical transport devices must change their default behavior. For instance a tripod turnstile which checks transport cards by default must let the user pass in case of emergency so that he can proceed to a safe zone for evacuation.

3.6.1.1 List of Actors

Actor	Type	Localization	Role
Smoke detector	Device		Fire detection
Emergency center	Human		Emergency management
Tripod turnstile	Device		User check

3.6.1.2 Storyboard

Step #	Actor	Event
1.	Smoke detector	Smoke is detected by the smoke detector.
2.	Smoke detector	The detector sprays water where the smoke has been detected.
3.	Emergency center	The alarm is set off.
4.	Tripod turnstile	The tripod turnstile does let the users pass.
5.	Emergency center	The center manages the crisis.
6.	Tripod turnstile	When the emergency is ended, the default turnstile behavior is restored.
7.	Emergency center	The emergency is closed.

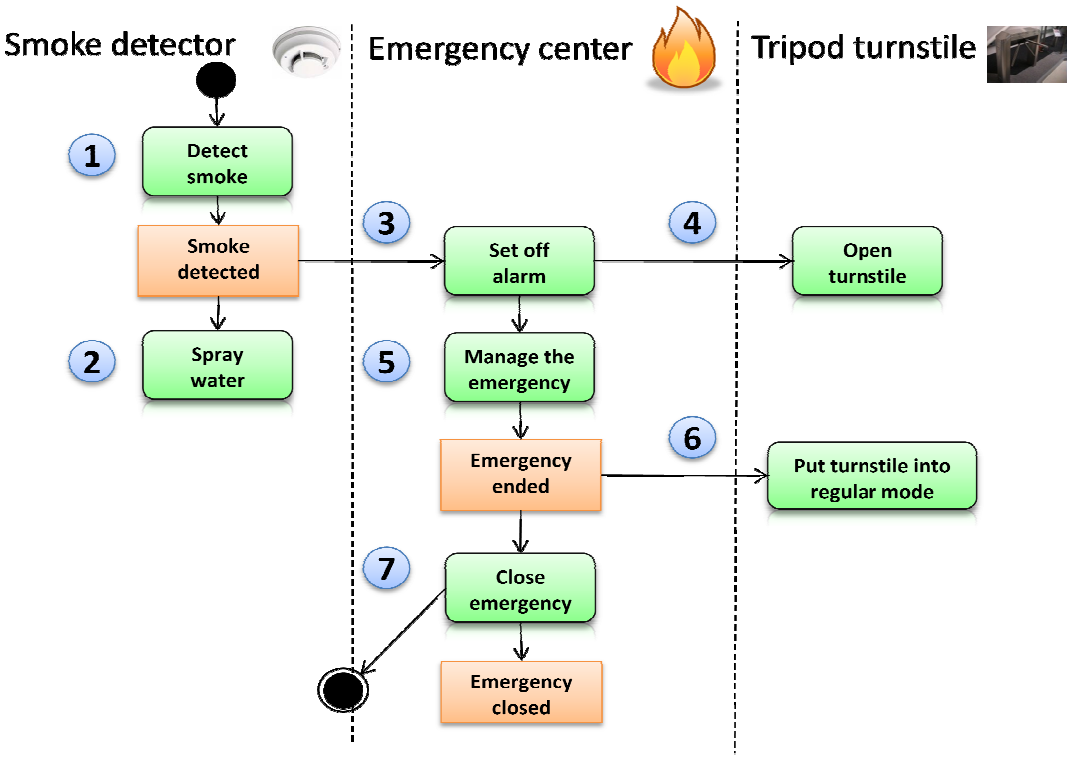

Figure 3-11: Fire emergency sequence diagram

3.6.1.3 Network Topology

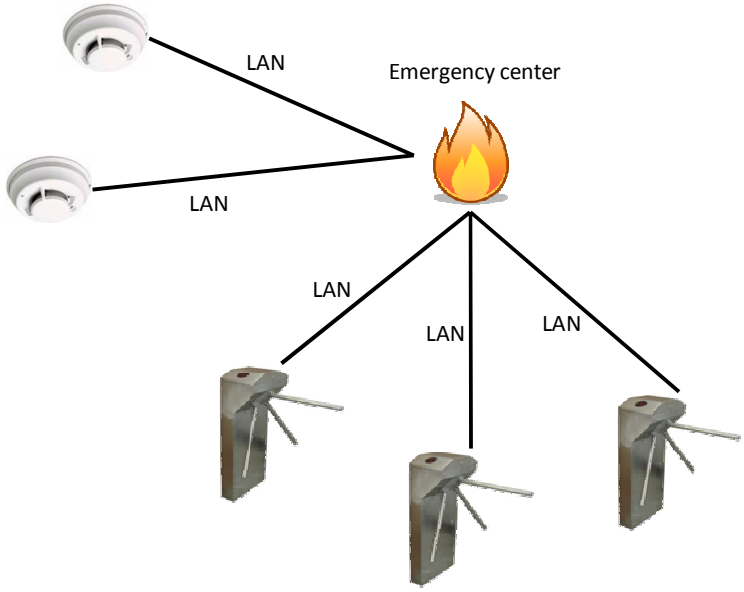

Figure 3-12: Network topology for the fire emergency system

3.6.2 Interoperability Issues Highlighted by the Scenario

3.6.2.1 Interaction Protocol Interoperability

The smoke detectors are bought from different manufacturers and are very unlikely to use the same communication protocols even though they are all IP based protocols. Similarly, the turnstiles also use different IP based protocols. The challenge of this scenario is the diversity of protocols over IP.

3.6.2.2 Data Interoperability

As with the many different protocols over IP, the devices involved in this scenario use many different data models.

3.6.3 Mapping to CONNECT Issues

3.6.3.1 Learning Requirements

It is assumed that the protocols used by each device are known and thus no learning is required.

3.6.3.2 Connector Synthesis Requirements

A lot of connectors need to be synthesized in this scenario. The main challenge is that as many of the protocols over IP used by those devices are proprietary protocols, there is no certainty that they are compatible.

3.6.3.3 Dependability and Security Requirements

The scenario does not highlight any dependability or security requirements.

4 Telco Web 2.0 & Cloud computing

4.1 Scenario: Online user reputation enabler

Contributors: Massimo Paolucci, Bertrand Souville

4.1.1 Description

It is assumed in this scenario that mobile users upload their videos to YouTube and can share their presence information (e.g. geographical information) using the RCS presence functionality deployed by mobile network operators. The network operator monitors the reputation of its subscribers by retrieving community-based feedback information at the local level (i.e. closed to the current location of the subscribers) from the video server. For instance, a subscriber with high reputation would have a good average rating score for his YouTube videos.

Another community-based platform Flickr starts providing video sharing services and a new connector is then synthesized automatically so that the network operator is able to monitor the reputation of its subscribers for Flickr as well.

4.1.1.1 List of Actors

Actor	Type	Localization	Role
Mobile user	Human	Connected to cellular network	Video content producer
RCS Presence Server	Service	IMS domain of network operator	Provides presence information of users
IMS Application Server	Service	IMS domain of network operator	Implements client interface to Web 2.0 services
YouTube Server	Service	Internet	Community-based video server
Flickr Server	Service	Internet	Community-based media server

4.1.1.2 Storyboard

Step #	Actor	Event
1.	**RCS Presence Server**	RCS Presence Server notifies to the IMS Application Server of changes in the presence information of users
2.	**IMS Application Server**	Geo-query for Flickr videos at the local level
3.	**IMS Application Server**	Retrieves community-based feedback information for each video

4.1.1.3 Network Topology

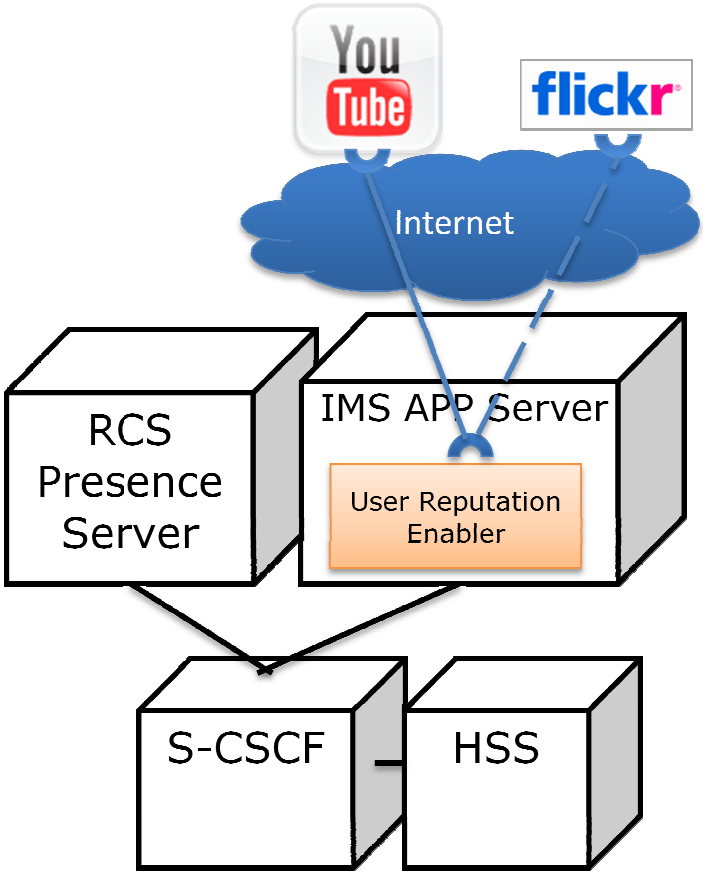

Figure 4-1: Network topology of the combined telecom and web 2.0 services use case

4.1.2 Interoperability Issues Highlighted by the Scenario

4.1.2.1 Interaction Protocol Interoperability

Both community-based servers provide similar functionalities but use different APIs and protocols (e.g. REST versus SOAP). Figure 4-2 shows an example of correct interaction behaviors for both systems using Labeled Transition Systems. A mediating connector is therefore required so that an existing client is able to interoperate with the Flickr media server.

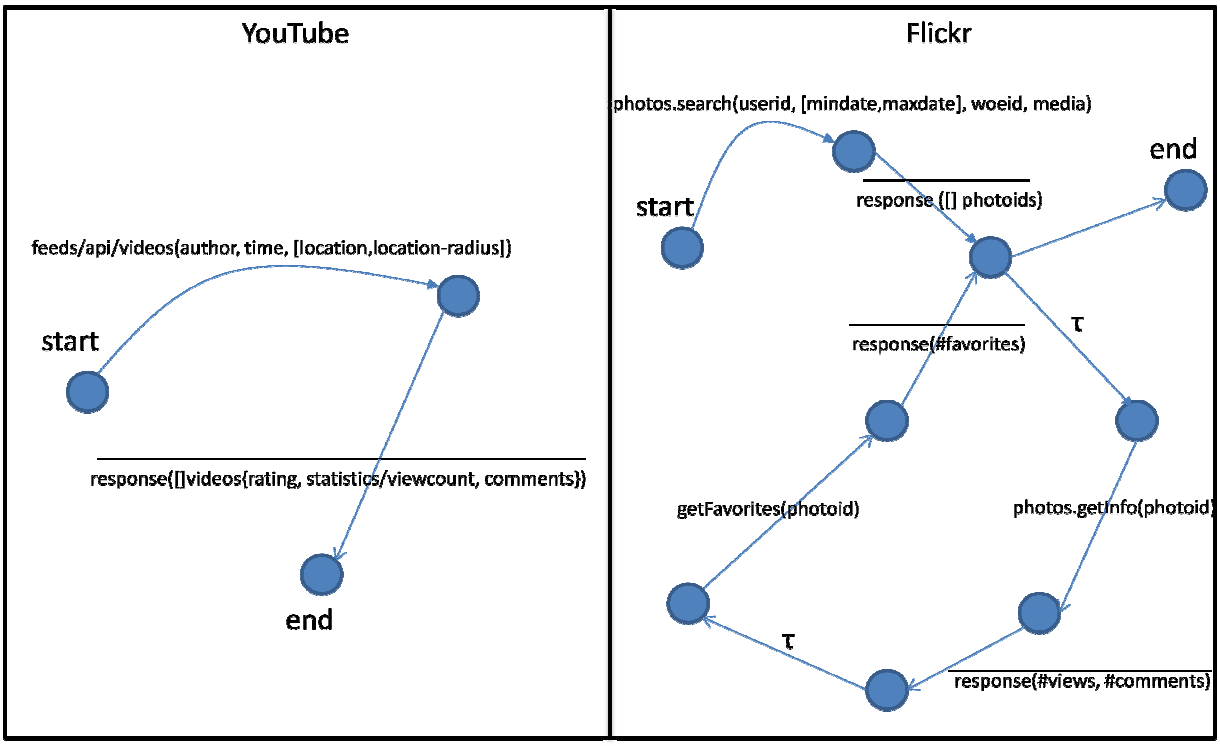

Figure 4-2: Labeled transition system for the YouTube and Flickr systems

4.1.2.2 Data Interoperability

This scenario also stresses data and semantic interoperability issues. By integrating systems from telecom and Internet domain, it cannot be assumed that all systems agree on a common standard (e.g. Presence Data Model standardized at the Open Mobile Alliance) and therefore there is a need to provide translation/mapping mechanisms to exchange data between systems that use different data models. Besides, heterogeneity problems also arise at the semantic level as geographic concepts provided by those systems may have different meanings. Semantic interoperability may be then achieved by defining a set of relationships between concepts that correspond semantically to each other.

4.1.3 Mapping to CONNECT Issues

4.1.3.1 Learning Requirements

Flickr provides reflection APIs that define the service method signatures (e.g. name, input arguments) currently supported by the platform. Figure 4-3 shows an example of such a service method signature.

```
<method name=”flickr.photos.getFavorites” …>
  <description>
      Returns the list of people who have favorited a given photo.
  </description>
  …
</method>
<arguments>
    <argument name="api_key" optional="0">
    Your API application key…
    </argument>
    <argument name="photo_id" optional="0">
    The ID of the photo to fetch the favoriters list for.
    </argument>
…
</arguments>
```

Figure 4-3: Example for the flickr.photos.getFavorites method signature

In order to access this information (required for the actual synthesis of the connector), a learning component may be beforehand used to learn the protocols supported by Flickr (e.g. REST).

4.1.3.2 Connector Synthesis Requirements

As already explained in section 4.1.2.1, a mediating connector is required between the existing User Reputation Enabler client deployed within the Telecom domain and the Flickr server.

4.1.3.3 Dependability and Security Requirements

Dependability and security requirements will be considered in a revised version of this use case. For instance, it would be interesting to incorporate QoS constraints or privacy requirements that shall be respected by the mediating connector.

5 Interoperability challenges: a summary

These scenarios identify the following important dimensions of interoperability that must be resolved by CONNECT solutions in order to ensure that spontaneously interacting peers can interoperate with one another at runtime:

- *Application Level Interoperability*. Application interfaces may be heterogeneous and peers may be implemented using different application logic e.g. different ordering of interchangeable operations.
- *Interaction Protocol Interoperability*. Different middleware protocols (e.g. different service discovery and interaction protocols) may be used to implement the application.
- *Data Level interoperability.* The data shared between peers may be formatted using different representations.
- *Interoperability of Non-functional Properties.* Peers may have particular non-functional properties e.g. latency of message delivery, dependability measures and security requirements that must be resolved with respect to the dynamically connected peer.

The table below summarizes these dimensions with respect to the scenarios (where N/A is used, it states that the scenario text does not identify the issue, as opposed to it not being a problem in the particular scenario). It is clear that these dimensions occur in a number of scenarios and hence, CONNECT can make significant strides forward to resolve the challenges in developing such open and dynamic applications.

Experiment Scenario	Interoperability Dimension	Description
Flood Prediction and Monitoring	Application Level	N/A
	Interaction Protocol Level	The middleware deployed on the different sensors uses different communication abstractions: a query request paradigm versus an event notification paradigm. Two separate middleware interaction protocols deploy this behavior.
	Data Level	Depth and flow rate measures are given in different units.
	Non-functional Properties	N/A
Road Tunnel Accident	Application Level	N/A
	Interaction Protocol Level	Three different middleware protocols employ different communication paradigms to underpin system behavior e.g. a messaging middleware, a tuple space, and an event notification service. These cannot interoperate in order to provide an integrated solution.
	Data Level	N/A
	Non-functional Properties	N/A
Stadium	Application Level	N/A

Warning System	Interaction Protocol Level	The stadium warning system uses a Publish-Subscribe middleware protocol, whereas the system on the user's device is based on a SOAP-based middleware. These two protocols use differing interaction patterns, leading to interoperability problems which must be addressed.
	Data Level	N/A
	Non-functional Properties	The connection between the heterogeneous systems must meet the original requirements of the warning system application in terms of latency of the warning message delivery.
Distributed Marketplace (Popcorn Scenario)	Application Level	The peers are implemented using different logic. This might include different ordering of interchangeable operations, or using a single higher-level operation instead of several lower level ones.
	Interaction Protocol Level	In one instance of the scenario, the Merchant's system is based on SSDP(UPnP) for discovery, and SOAP for request handling. The Consumer's system is based on a LIME implementation of Tuple Spaces. These are very different middleware with completely different interaction patterns, and need reconciliation.
	Data Level	The Merchant and the Consumer may assume different currencies while representing costs. This interoperability problem must be addressed for proper functioning of the application.
	Non-functional Properties	Different authentication and authorization protocols may be employed by the two peers where payment is involved. The heterogeneity between security mechanisms must be resolved.
Airport Boarding Cards	Application Level	N/A
	Interaction Protocol Level	The two peers use different discovery protocols (SDP and Bonjour) and hence cannot find one another. The interaction protocol where the client sends a request to the printers is different at each peer i.e. the client sends a BPP request and the printer server uses LPR.
	Data Level	The two interaction protocols have different data representations for important printer job characteristics e.g. paper size, number of copies. The server must be able to understand the data sent from the client.
	Non-functional Properties	N/A
Car Parking	Application Level	N/A
	Interaction Protocol Level	N/A

	Data Level	This scenario points out two types of data interoperability problems: i) how to adapt a car ID to an valid ID with regards to the parking authority and ii) how to transform a location ID to a parking ID. While the first transformation is rather simple to deal with, the second transformation is way more complex as it requires a representation of the location of the parking lot in some coordinate structure and then potentially a second mapping from the coordinate system used by the mobile to the coordinate system used by the mapping.
	Non-functional Properties	N/A
Card checking	Application Level	N/A
	Interaction Protocol Level	The concentrator manages a network of RFID (13) card readers, defining the type of cards to index and the actions to take. Protocol interoperability is required to allow communication between the reader network controller and the readers on an IP network.
	Data Level	Data from the card readers need to be adapted and transformed into the concentrator exchange format.
	Non-functional Properties	N/A
Card order	Application Level	N/A
	Interaction Protocol Level	The communications between the actors use various protocols depending on the age of the applications that run on the Accounting system and the Card ordering system. Communications are not trivial as there is a mix of protocols like IIOP (14), REST (15) and SOAP (16).
	Data Level	It cannot be assumed that all the systems use the same data models. The information provided by the customer are very likely stored in different database schemas by the Accounting and the Card ordering systems as well as the Card manufacturer.
	Non-functional Properties	Heterogeneity of authorization and authentication mechanisms between connecting peers.
Ticket order	Application Level	N/A
	Interaction Protocol Level	Various communication protocols are used in this scenario. Applications that interact with the customer will preferably use REST or SOAP message or basic HTTP forms. Communications between the servers will preferably use SOAP or JMS (17).
	Data Level	The systems use different data models. There could be as much data models as the number of actors making it difficult for the connectors to adapt to all of them.

	Non-functional Properties	Heterogeneity of authorization and authentication mechanisms between connecting peers.
Fire emergency	Application Level	N/A
	Interaction Protocol Level	The smoke detectors are bought from different manufacturers and are very unlikely to use the same communication protocols even though they are all IP based protocols. Similarly, the turnstiles also use different IP protocols. The challenge of this scenario is the diversity of IP protocols.
	Data Level	As with the many different IP protocols, the devices involved in this scenario use many different data models.
	Non-functional Properties	N/A
Online User Reputation Enabler	Application Level	Both systems (YouTube and Flickr) provide similar functionalities but use different interfaces and data models. The challenge is to enable automated mediation at the application-layer level (See D3.1 (18) for details).
	Interaction Protocol Level	Both systems use REST i.e. http so there is no interaction protocol interoperability problems.
	Data Level	The challenge is on the one hand to define data mapping/translation mechanisms that identify attributes that carry the same meaning. On the other hand, at the semantic level, the matching of ontological concepts assumed to be done at run time in CONNECT remains an open problem (See D1.1 (1) for details).
	Non-functional Properties	Maintenance of QoS constraints and privacy of peers in the connected system.

6 Conclusion

The scenarios detailed in this document emphasize the issues that one might deal with when connecting networked systems together and the importance of connectivity solutions such as CONNECT.

The collected scenarios are complementary and cover the most usual situations in three different domains (Ubiquitous computing, Daily life support and Telco Web2.0 & Cloud computing) where interoperability is very often a key issue.

The range of interoperability issues highlighted by the presented scenarios goes from issues with different communication abstractions (request/reply vs. events) to issues with data transformations.

As the project progresses, they are very likely to evolve and be refined in the future to reflect the new and important directions that the CONNECT solutions address.

7 References

1. CONNECT**.** *D1.1 - Initial* CONNECT *Architecture.* 2010.

2. Zigbee Alliance Homepage. [Online] http://www.zigbee.org/.

3. Bluetooth Core System Architecture. [Online] http://bluetooth.com/Bluetooth/Technology/Works/Core_System_Architecture.htm.

4. **B. Jiao, S. Son, and J. Stankovic.** *GEM: Generic event service middleware for wireless sensor networks.* San Diego, CA, USA : 2nd International Workshop on Networked Sensing Systems (INSS), June 2005.

5. **Madden, S. R., Franklin, M. J., Hellerstein, J. M., and Hong, W.** *TinyDB: an acquisitional query processing system for sensor networks.* s.l. : ACM Transactions on Database Systems, March 2005. pp. 122-173. Vol. 30 Issue 1.

6. **A. Carzaniga, D.S. Rosenblum, and A.L. Wolf.** *Design and Evaluation of a Wide-Area Event Notification Service.* s.l. : ACM Transactions on Computer Systems, August 2001. pp. 332-383. Vol. 19 Issue 3.

7. *RUNES (Reconfigurable Ubiquitous Networked Embedded Systems) IST Project.* [Online] http://ist-runes.org/.

8. **Zhang, M and Wolf, R.** *Border Node Based Routing Protocol for VANETs in Sparse and Rural Areas.* Washington D.C. : IEEE Globecom Autonet Workshop, November 2007. pp. 1-7.

9. **Murphy, A. L., Picco, G. P., and Roman, G.** *LIME: A coordination model and middleware supporting mobility of hosts and agents.* s.l. : ACM Transactions on Software Engineering Methodology, July 2006. pp. 279-328. Vol. 15 Issue 3.

10. **Cheshire, S. and Krochmal, M.** Multicast DNS. [Online] September 2009. http://files.multicastdns.org/draft-cheshire-dnsext-multicastdns.txt.

11. **McLaughlinIII, L.** Line Printer Daemon Protocol. [Online] August 1990. http://tools.ietf.org/html/rfc1179.

12. iRDA, OBEX specification. [Online] http://www.irda.org/displaycommon.cfm?an=1&subarticlenbr=7.

13. *RFID Learning Guide.* [Online] http://searchmobilecomputing.techtarget.com/news/article/0,289142,sid40_gci959999,00.html#specs.

14. CORBA IIOP Specification. [Online] http://www.omg.org/technology/documents/formal/corba_iiop.htm.

15. **Elkstein, M. (Dr.).** Learn REST. [Online] http://rest.elkstein.org/2008/02/what-is-rest.html.

16. SOAP Specifications. [Online] http://www.w3.org/TR/soap/.

17. Java Message Service Specification. [Online] http://java.sun.com/products/jms/docs.html.

18. CONNECT**.** *D3.1 - Modeling of application and middleware layer interaction protocols.* 2010.